Optimization of Gaussian Surface Calculations and Extension to Solvent-Accessible Surface Areas

JÖRG WEISER, PETER S. SHENKIN, W. CLARK STILL
Department of Chemistry, Columbia University, New York, New York 10027

Received 23 October 1998; accepted 22 December 1998

ABSTRACT: We explored the use of several breadth-first and depth-first algorithms for the computation of Gaussian atomic and molecular surface areas. Our results for whole-molecule van der Waals surface areas (vdWSAs) were 10 times more accurate in relative error, relative to actual hard-sphere areas, than those reported by earlier workers. We were also able to extend the method to the computation of solvent-accessible surface areas (SASAs). This was made possible by an appropriate combination of algorithms, parameters, and preprocessing steps. For united-atom 3app, a 2366-atom protein, we obtained an average absolute atomic error of 1.16 $Å^2$ with respect to the hard-sphere atomic SASA results in 7 s of CPU time on an R10000/194 MHz processor. Speed and accuracy were both optimized for SASA by the use of neighbor-list reduction (NLR), buried-atom elimination (BAE), and a depth-first search of the tree of atomic intersections. Accuracy was further optimized by the application of atom type specific parameters to the raw Gaussian results. J Comput Chem 20: 688–703, 1999

Keywords: Gaussian shape; van der Waals surface areas (vdWSA); solvent-accessible surface area (SASA); neighbor-list reduction (NLR); buried atom elimination (BAE); breadth-first algorithm; depth-first algorithm

Correspondence to: J. Weiser; e-mail: joerg@still3.chem.columbia.edu
Contract/grant sponsor: Deutsche Forschungsgemeinschaft
Contract/grant sponsor: NSF; contract/grant number: CHE97-07870

CCC 0192-8651 / 99 / 070688-16

Introduction

Molecules are often represented as a set of overlapping spheres. The solvent-accessible surface area (SASA) of a molecule is widely used in describing the solvation of solutes and macromolecules. The SASA is defined as the surface traced by the center of a sphere rolled over the van der Waals surface (vdWSA). The free energy of aqueous solvation, at least for nonpolar molecules, is linearly related to SASA.[2] Therefore, many methods (some exact, some approximate) have been presented to calculate analytical atomic SASAs and their partial derivatives with respect to atomic coordinates.[3–21]

Grant and Pickup recently proposed an interesting description of molecular shape based on the representation of each atom's volume by a spherically symmetrical Gaussian density distribution.[22] The intersection volume of two or more atoms is approximated by the product of the atomic Gaussian distributions. Computations are simplified by the Gaussian product theorem, which prescribes how each such multiple product can be represented by a single Gaussian located at the appropriately weighted centroid of the contributing atoms' positions (Appendix A). An atomic area is obtained by taking the derivative of the total volume with respect to the radius of the atom in question, just as the area of a sphere is obtained by taking the derivative of its volume with respect to its radius. The authors showed that this method could be programmed and parameterized to allow rapid computation of reasonable approximations to hard-sphere vdW volumes and surface areas for macromolecules.[22–24] The method is particularly attractive because it readily affords first and second derivatives of these quantities with respect to nuclear position.

The total volume of a molecule is the sum of signed intersection volumes: the intersection volume is taken with a positive sign for odd orders and a negative sign for even orders.

$$V_{\text{tot}} = \sum_i V_i - \sum_{i<j} V_{ij} + \sum_{i<j<k} V_{ijk} - \sum_{i<j<k<l} V_{ijkl} + \cdots \quad (1)$$

When there are many spheres and their overlap is great, the number of intersection volumes that have to be calculated in order to evaluate eq. (1) can be enormous. The number of overlapping atom pairs grows by a factor of five to seven when a SASA rather than a vdWSA is computed.[25] This is because the vdW radii are augmented by the probe size, which is typically 1–2 Å, while the interatomic distances remain the same. Thus, the range of each sum, as well as the number of sums, grows dramatically.

To the best of our knowledge, there are no successful attempts in the literature to extend the Gaussian method to the computation of SASA. In their original Gaussian method article, Grant and Pickup made several suggestions for this extension and discussed some of the problems involved in doing so.[22] The main problem is that the product of two Gaussians is a poor approximation to a hard-sphere intersection volume when the spheres overlap greatly. For example, the intersection of two hard spheres of equal radius situated on the same center is the volume of one of them; however, the product of two identical Gaussian density distributions located on the same center is the square of one of them. Furthermore, the product of N spherical Gaussian distributions is another spherical Gaussian distribution, but the intersection of N hard spheres does not, in general, have spherical symmetry. Grant and Pickup suggested that the problem might be solved by replacing a single Gaussian on each atomic center by a sum of two or more Gaussians.[22] Out of concern for runtime performance, we did not follow this suggestion.* Instead, we concentrated on improving the parametrization of the method and on devising algorithms to perform the computation rapidly.

For performance optimization, we experimented with several breadth-first (BF) and depth-first (DF) methods for traversing the volume intersections. The literature to date[22–24] describes only the use of

* Suppose each atomic center is represented by n Gaussians and consider an intersection of N atoms. If the intersection is approximated by the product of the sums of the Gaussians on each center, then there will be n^N individual Gaussian products to evaluate for each such intersection. SASA requires intersections of order up to $N = 12$; if n is 2, such an intersection will require 2^{12} or about 4000 times more computation time than when n is 1. Even the computation of pairwise atomic intersections will be slower by a factor of 4. These estimates can be considered upper limits, because some pairs of Gaussians across intersecting atom pairs will have negligible products; further, it might be possible to represent some atom types as single Gaussians. It would appear, however, that using sums of several Gaussians on even some atomic centers is likely to considerably slow the evaluation of Gaussian SASAs.

BF methods in this regard. We also incorporated new methods for neighbor-list reduction (NLR)[25] and buried atom elimination (BAE, see Appendix B)[26] into the traversal and examined the effects on performance. Finally, we examined the utility of Gibson and Scheraga's method for eliminating the computation of spherical intersections of order greater than four.[27] These methods all apply only to the parsing and exploration of the terms in eq. (1). Because eq. (1) is used in a variety of methods for computing atom surfaces (e.g., hard sphere, as well as Gaussian), the results may apply to these other such methods and to the computation of Gaussian surfaces. To optimize agreement of Gaussian surfaces with hard-sphere results, we experimented with varying the Gaussian p parameter (see Appendix A). In addition, we found for both vdWSA and SASA that applying an atom type dependent multiplier to the raw Gaussian results afforded additional accuracy.

Seventeen compounds (Table I) of different sizes (11–2,366 atoms) and classes (small organics, proteins, DNA and various complexes) have been chosen as representative test cases for the calculation of Gaussian vdWSA and SASA.

Methods

PREPROCESSING STEPS

NL and NLR

Any two Gaussians, no matter how far apart, have a finite product. Thus, the full computation of Gaussian volume for n spheres according to eq. (1) must include intersections of all orders up through n. However, because the overlap is small if the spheres are far apart, Grant and Pickup[22] introduced a cutoff criterion based on hard-sphere radius. An intersection of n spheres is considered to have nonzero volume only if all pairs within the set are within the sum of the hard-sphere radii of each other. This defines, for each atom, a neighbor list $N(i)$, of all the other spheres that intersect with i. The introduction of a neighbor list not only decreases the ranges of the sums in eq. (1), but also truncates the list of sums; for example, if even one pair in an ensemble of n spheres does not intersect, no nth-order intersection can occur.

The number of intersection volumes can be further reduced by applying the recently reported

TABLE I.
Compounds.

chb:	3-Chloro-4-hydroxybenzoic acid (C_7 H_5 O_3 Cl) Complexed with protocatechuate 3,4-dioxygenase (Brookhaven entry 3pch) The inhibitor is observed in two binding sites. The first, which we used, is in the active site of each protomer (residue 550).
ctc:	7-Chlorotetracycline (C_{22} H_{23} N_2 O_8 Cl) Complexed with tetracycline repressor from *Escherichia coli* (Brookhaven entry 2tct)
sip:	Sipholenol-A monoacetate (C_{32} H_{54} O_5); global MM3(92) energy minimum[31]
nmx:	Nitromethyldethia coenzyme A (C_{22} H_{37} N_8 O_{18} P_3) Complexed with chicken citrate synthase complex (Brookhaven entry 1amz) Two conformers of the side chain are given in the pdb file. We took conformer A.
1van:	Vancomycin complex with L-Lys-D-Ala-D-Ala (theoretical model)
1crn:	Crambin
103d:	DNA (5′-D(*GP*TP*GP*GP*AP*AP*TP*GP*GP*AP*AP*C)-3′) (antiparallel DNA duplex; human centromere repeat)
2ins:	Insulin from bovine (*Bos taurus*)
163d:	Rev responsive element (RBE, 30 ribonucleotide fragment) complexed with HIV rev protein (residues 34–50)
1lz1:	Human lysozyme
2stw:	Human ETS1 / DNA complex
2tra:	Transfer ribonucleic acid (yeast, Asp) with spermine (C_{10} H_{26} N_4). We took conformer A.
1sbg:	HIV-1 protease complexed with the inhibitor SB203386
5tra:	Transfer ribonucleic acid (yeast). We did not take into account the metal atom in the pdb file (M7, atom 255).
1inc:	Porcine pancreatic elastase complex with benzoxazinone inhibitor (C_{17} H_{22} N_2 O_4 Cl)
1kvd:	Killer toxin from halotolerant yeast
3app:	Fungus acid proteinase (penicillopepsin)

All acronyms containing four characters are Brookhaven entries.[30]

NLR method. Thus, $N(i)$ can usually be considerably shortened by removal of atoms whose elimination cannot affect the exposed surface of i. For vdWSA, neighbor lists are reduced to about 90% (united atom) or 60% (all atom) of their original sizes; for SASA, neighbor lists are reduced to about 37% (united atom) or 27% (all-atom) of their original sizes.[25]

When NLR is used, the total volume computed using eq. (1) will be incorrect, because NLR culls only pairwise intersections. However, exposed surface areas will be correctly computed. The reason is that when we consider intersections contained within a culled pairwise intersection (e.g., $V_{1\text{-}2\text{-}3}$ when $V_{1\text{-}3}$ and $V_{3\text{-}1}$ have been culled), we are careful to compute surface contributions only for atoms not in the culled intersections (in this example only the radial derivative of $V_{1\text{-}2\text{-}3}$ with respect to atom 2 is calculated). We mention this as a cautionary note to users who might wish to apply the NLR variants of the methods described here to molecular volume computations.

BAE

Up to about half of the atoms in biopolymers are inaccessible to solvent. The quick, approximate BAE method was developed for the detection of such atoms.[26] The method makes use of a Gaussian function, in the spirit of Stouten et al.,[28] to calculate the neighbor density in four tetrahedral directions in 3-dimensional space, sometimes twice with different orientations (see Appendix B). If the neighbor density is high in all directions, the atom is considered buried. The BAE method, although not exact, provides a good mixture of accuracy, efficiency, and speed. It is used here as a preprocessing step for SASA calculations. BAE is not, however, useful in the computation of vdWSAs, because no atoms are completely buried in this situation, at least when united atoms are used.

Elimination of Higher Order Intersection Volumes

Gibson and Scheraga (GS) showed that in 3 dimensions eq. (1) can be expressed as a sum of signed intersections of fourth order or lower; higher order intersections can be expressed as sums of intersections of lower order.[27] However, composition of the sums requires examination of all higher order intersections.[29] We used the present version of their code in which they investigate overlaps up to order 8. Determination of the surviving intersections and their number, represented by the integer coefficients n_i, n_{ij}, n_{ijk}, and n_{ijkl}, was used as a preprocessing step (GS) and the Gaussian method was subsequently performed according to eq. (2).

$$V_{\text{tot}} = \sum_i n_i V_i - \sum_{i<j} n_{ij} V_{ij} + \sum_{i<j<k} n_{ijk} V_{ijk} - \sum_{i<j<k<l} n_{ijkl} V_{ijkl}. \quad (2)$$

This is much in the spirit of the use by Grant and Pickup[22] of hard-sphere criteria to cull the list of intersections prior to performing Gaussian surface calculations.

BF VERSUS DF SEARCH OF INTERSECTION VOLUMES

BF

The BF approach calculates eq. (1) order by order: all spheres of order 1 are summed; then all intersections of order 2 are summed, and so on. Grant and Pickup suggested stopping when the last intersection order examined makes only a negligible contribution to the total surface. They reported that, for vdWSA, only terms up to sixth order need be included.[22] So far this has been the method reported for all Gaussian surface calculations in the literature.[22–24]

We optimized the BF method by applying it atom by atom: we first computed all of atom 1's first-order intersections, then all of its second-order intersections, and so on, stopping whenever the last set of intersections contributed less than the BF surface-area cutoff criterion, ε_S, to the surface area of atom 1. Then we proceeded to atom 2 and so on. The atom by atom approach allows the maximum BF order to differ for each atom.

DF

In contrast to BF, the DF method does not compute eq. (1) in an intersection order manner. Like BF, this algorithm starts with atom 1 at first order, but increments the intersection order after every calculation of an intersection volume by adding a new sphere from the neighbor list to the previous ensemble of intersecting spheres. Atomic surface contributions are calculated at each stage as radial derivatives of the intersection volume (see Appendix A). If an intersection volume is below ε_V, the volume cutoff criterion for DF, no surface contributions of this volume intersection

will be calculated and no higher order intersections will be derived from this one. Instead, we will ''back up'' and construct another intersection of the same order from the parent of lower order by discarding the new neighbor just added and replacing it with a different neighbor. On the other hand, if the volume of a new intersection is above ε_V, DF continues by incrementing the intersection order as described above.

The compound of Figure 1 serves as an example for the DF algorithm. Suppose atoms 2, 3, 4, and 5 are on the neighbor list of atom 1. In the DF method we would calculate $V_1, V_{1\text{-}2}, V_{1\text{-}2\text{-}3}$, in that order. Next we would ordinarily compute $V_{1\text{-}2\text{-}3\text{-}4}$, but suppose that $V_{1\text{-}2\text{-}3}$ is below ε_V. We would then instead compute $V_{1\text{-}2\text{-}4}, V_{1\text{-}2\text{-}4\text{-}5}$, (suppose below ε_V), $V_{1\text{-}2\text{-}5}$ (suppose below ε_V); then we have no more neighbors of 1, so we would continue with $V_{1\text{-}3}, V_{1\text{-}3\text{-}4}$, and so on.

The DF method avoids calculating all possible intersections of every ensemble at any order. In the above example, only a single fourth-order intersection was visited; in the BF method all would have been visited, if any was, at least for a given central atom. We might expect, therefore, that DF will be faster than BF, especially when higher-order intersections are to be computed as for SASA.

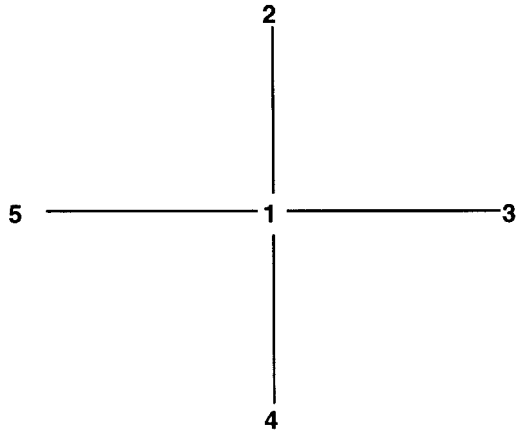

FIGURE 1. Hypothetical model compound. The compound in this figure has the following reduced and unreduced neighbor list:

Atom	Unreduced Neighbors	Reduced Neighbors
1	2, 3, 4, 5	2, 3, 4, 5
2	1, 3, 4, 5	1, 3, 5
3	1, 2, 4, 5	1, 2, 4
4	1, 2, 3, 5	1, 3, 5
5	1, 2, 3, 4	1, 2, 4

THREE VARIATIONS OF TWO BASIC METHODS

There are several ways in which to select when and how surfaces are derived from the intersection volumes visited by the BF and DF methods. This, and the interaction of this question with the use of the preprocessing optimizations, introduces several variations into either method.

Variation 1

All intersection volumes are calculated for every atom i and the radial derivatives of every intersection volume are calculated with respect to atom i only. Thus, DF-1 would involve visiting intersection $V_{1\text{-}2}$ during the computation of the exposed surface area of atom 1 and then later visiting $V_{2\text{-}1}$, which is actually the same intersection, during the computation of the surface area of atom 2. This method works properly whether or not NLR is first performed.

Variation 2

It would be desirable to not have to visit intersection $V_{1\text{-}2}$ twice, first as such, then again as $V_{2\text{-}1}$. However, avoiding this duplication introduces complications when NLR is carried out.

Variation 2a. Without NLR, only $V_{i\cdots n}$, where $i < \cdots < n$, are visited. Thus, after computing the $V_{1\text{-}2\text{-}\cdots n}$ intersections and V_2, we would proceed not to $V_{2\text{-}1}$ but to $V_{2\text{-}3}$. This guarantees that every intersection will be visited only once. As each intersection is visited, the surfaces of all atoms that contribute to the intersection are computed.

Variation 2b. Variation 2a is not directly applicable when NLR is applied. Suppose, for example, that atoms 1 and 3 are on the neighbor list of 2, but that 3 and 1 are not on each others' neighbor lists. Then, after visiting $V_{1\text{-}2}$, we would have no way to proceed to $V_{1\text{-}2\text{-}3}$; but we need $V_{1\text{-}2\text{-}3}$ to obtain an area term for atom 2. Variation 2a supplies no way to visit this intersection.

In variation 2b we use the unreduced neighbor list to construct the intersections that we visit; in this regard, the algorithm is identical to 2a. However, at each stage we compute surface area contri-

butions only for those atoms whose neighbor lists include all the other atoms comprising the intersection.

Thus, 2a is better than variation 1 when NLR is not in use. However, with NLR, it is not clear whether 2b (with its elaborate checking) will be more efficient than variation 1; it is not even obvious, *a priori*, that variation 2b with NLR will be more efficient than variation 2a without NLR.

Variation 3

Variation 3 is potentially useful only when NLR is in use, and it consists of two parts. In the first part, we visit $V_{i\ldots n}$, where $i < \cdots < n$ are taken from the reduced neighbor list of i. A surface contribution is always computed for i, because we know that all the other intersections are on i's neighbor list; however, for the other atoms comprising the intersection, we compute a surface contribution only if all the other atoms comprising the intersection are on its (reduced) neighbor list. As pointed out in the discussion of variation 2b, such a procedure will neglect to visit certain necessary intersections. Therefore, in the second stage of variation 3 we locate and visit the intersections we missed during the first stage.

NLR works symmetrically: if atom j is removed from the neighbor list of i, then atom i will also be removed from the neighbor list of j. This means, for example, that if the first part of the algorithm passes over $V_{1\text{-}2}$, we know we do not have to visit $V_{2\text{-}1}$. Thus, the second stage of variation 3 need only locate ''missing'' intersections of order 3 or greater. Also, since the first stages of a search begin with atom 1 and operate based on the neighbor list of 1, the leading index in the missing intersections will be at least two. In stage 2, therefore, we start with atom 2, then proceed to atom 3 and so on. Finally, since the first stage explores all intersections with indices in increasing order, at least one index in each missing intersection will be less than the leading index; we will skip any others in stage 2.

These features are exhibited by the example discussed in connection with variation 2b. There the missing intersection was $V_{1\text{-}2\text{-}3}$, which would not have been encountered in the course of the first stage of variation 3; however, in the second stage, it would be encountered as $V_{2\text{-}1\text{-}3}$. Following a check to be described in the next paragraph, the pair list of each atom in the intersection is examined to see which atoms' neighbor lists include all the other atoms. For this particular example, this would be true only for atom 2. However this might be, surface contributions would be computed only for these atoms.

Before doing the surface calculations, however, another check must be done. We have to make sure that a given intersection encountered in stage 2 has not been previously used to compute surfaces either in stage 1 or earlier in stage 2. Suppose the list of indices of an intersection encountered in stage 2 is $m\text{–}o\text{–}\cdots\text{–}q$. Suppose that $o\text{–}\cdots\text{–}q$ are in increasing order. Then we know that $m > 1$, $o < m$, and $o\text{–}\cdots\text{–}q$ are on the reduced neighbor list of m. If for any of the other atoms with indices less than m it is also true that all the atoms in the intersection are on its reduced pair list, then we will have used this intersection earlier to compute surfaces and we should not do so now.

To give a concrete example, suppose we find that atoms 1 and 7 are on $N(3)$, the reduced neighbor list of atom 3. Then, during stage 2 we will encounter $V_{3\text{-}1\text{-}7}$. We would first check to see if 3 and 7 are on $N(1)$; if so, this intersection would have been used to compute surfaces in stage 1 and we would ignore it. Otherwise, because no other atoms in the intersection have values less than 3, we know that we need to compute at least a surface contribution to atom 3 and possibly a contribution to atom 7 (but not one to atom 1). We would compute 3's surface contribution and, if both 1 and 3 were on $N(7)$, a contribution to 7's surface as well. Later in stage 2 we might or might not encounter the 7-1-3 intersection; but if we did, we would discover during the preliminary check that all appropriate surface contributions were computed earlier and we would avoid even the computation of the $V_{7\text{-}1\text{-}3}$ Gaussian product.

Although the checking procedure in variation 3 is quite complex and this variation visits many intersections multiple times, it avoids computing the Gaussian product more than the absolutely minimal number of times necessary when NLR is in use. Thus, if the cost of computing the product is high relative to tree traversal and checking, this procedure would be the most efficient.

PARAMETERS

For BF searches, the lower the value of ε_S, the more accurate the results, down to some critical value, beneath which the accuracy remains constant. This critical value was the one used.

For DF searches, except where otherwise indicated, we lowered ε_V until the results were at

least as accurate as the BF results, then kept ε_V at this value.

For every combination of preprocessing steps, an optimal set of parameters was obtained in the following manner. A range of p parameters [eq. (A.5), Appendix A] was explored. For each value of p, the atomic Gaussian surface areas were determined and linear regression (assuming a zero intercept) was used to determine the best multipliers for our predefined atom types (based on atomic number, hybridization, and number of bonded neighbors[21]). The regression was carried out over the entire test set. In the end, we selected the value of p (together with the corresponding atom type based multipliers) that gave the least average unsigned error in atomic surface areas for 3app, on which we concentrate most of our analysis. Selecting p based on the least error over the entire test set would, if anything, have made the overall results better and the 3app results worse; however, we suspect that in fact the same parameters would have been chosen.

OTHER CONSIDERATIONS

BAE introduces further complications. When variation 1 is used, the computation of all intersection volumes of buried atoms is skipped completely. In variations 2 and 3, only the radial derivatives with respect to buried atoms are skipped; all intersections including buried atoms cannot be eliminated, because buried atoms occlude the surface on partly exposed atoms. Given the simplicity of variation 1 and the culling that BAE provides, it is at least possible that variation 1 might be most efficient when BAE is in use, even if some other variation might be most efficient otherwise.

The Gaussian method occasionally produces small negative surface areas for individual atoms. When these were encountered, they were simply set to zero.

Results

For all surface computations, we used the same atomic radii as Grant and Pickup[22] (Table II) and a solvent-probe radius of 1.4 Å. We studied the effect of the various algorithms and preprocessing steps on CPU time and accuracy of Gaussian united-atom vdWSA (Table III), all-atom vdWSA (Table IV), and united-atom SASA (Table V), taking penicillopepsin (3app), the largest compound in our test set, as representative test case. Using the same parameters (Table VI), we calculated the united-atom vdWSAs (Table VII) and the united-atom SASAs (Table VIII) of all 17 compounds listed in Table I. Atomic coordinates for molecules in the data set are published.[30,31] To build all-atom 3app, hydrogens were constructed on the published heavy atom backbone with standard bond lengths and angles by using the MacroModel program.[32] Accurate numerical surface areas were also computed using MacroModel/BatchMin. Most parameters were chosen to optimize one aspect or another of the results with 3app; thus, the reported values probably represent an overestimate of the accuracy for this compound and an underestimate of the accuracy for the other compounds. However, we expect that this effect is minor.

TABLE II.
Atomic van der Waals Radii.

Element	Radius (Å)
H	1.00
C	1.70
N	1.65
O	1.60
S	1.90
P	1.90
Cl	1.80

All calculations were performed on an SGI R10000/194 MHz processor (Power Onyx), using Fortran code optimized at the -n32 -mips3 -O3 level.

GAUSSIAN vdWSA

Table III shows that for any algorithmic variation, the DF method is faster than the corresponding BF method; however, the timings and accuracies are similar for all methods.

For united-atom 3app without NLR, the fastest method is DF-2a. Introduction of NLR naturally speeds up the computation of Gaussians [i.e., the time spent evaluating eq. (1)] in variation 1 but slows it down in variation 2, because the necessary additional checking costs more CPU time than a reduced neighbor list saves. NLR also worsens the accuracy (in terms of average absolute atomic error, which is used as the chief accuracy criterion throughout). With NLR, variation 3 is faster than variation 1; thus, the additional checking pays for itself. The CPU time for the Gaussian surface com-

TABLE III.

vdWSA Calculations of Penicillopepsin (3app, United Atom, 2366 Atoms, Numerical vdWSA = 30404 Å^2).

Pre- or Postprocessing Step	p[a]	CPU Pre- or Postprocessing Step(s) (s)	CPU Gaussian Calculations (s)			Total CPU (s)			vdWSA (Å^2) (% Diff. from Numer. Result)			Ave. Abs. Atomic Error (Å^2)		
			Variation	GS	BF	DF	GS	BF	DF	GS	BF	DF	GS	BF	DF
GS (pre)	2.50	1.468		0.043			1.511			29927 (1.6)			0.70		
NL (pre)	2.60	0.227	*V*1		0.247	0.147		0.474	0.374		29597 (2.7)	29723 (2.2)		0.61	0.61
			*V*2a		0.098	0.075		0.325	0.302		29581 (2.7)	29723 (2.2)		0.61	0.61
NLR (pre)[b]	2.60	0.303	*V*1		0.186	0.117		0.489	0.420		29444 (3.2)	29535 (2.9)		0.67	0.68
			*V*2b		0.104	0.081		0.407	0.384		29428 (3.2)	29535 (2.9)		0.68	0.68
			*V*3		0.141	0.111		0.444	0.414		29428 (3.2)	29535 (2.9)		0.68	0.68
Fitting (post)[c]	2.60	0.227	*V*2a			0.075			0.302			30430 (0.1)			0.30

Preprocessing step: NL only. The following cutoffs were used: BF, $\varepsilon_S = 0.005$ Å^2; DF, $\varepsilon_V = 0.04$ Å^3.
[a] See eq. (A.5).
[b] Including CPU time for NL.
[c] Using atom type dependent parameters.

TABLE IV.

vdWSA Calculations of Penicillopepsin (3app_H, All Atom, 4550 Atoms, Numerical vdWSA = 32253 Å^2).

Pre- or Postprocessing Step	p[a]	CPU Pre- or Postprocessing Step(s) (s)	CPU Gaussian Calculations (s)			Total CPU (s)			vdWSA (Å^2) (% Diff. from Numer. Result)			Ave. Abs. Atomic Error (Å^2)		
			Variation	GS	BF	DF	GS	BF	DF	GS	BF	DF	GS	BF	DF
GS (pre)	2.30	10.301		0.062			10.363			32380 (0.4)			0.59		
NL (pre)	2.65	0.881	*V*1		0.723	0.406		1.604	1.287		31004 (3.9)	31751 (1.6)		0.66	0.51
			*V*2a		0.303	0.202		1.184	1.083		31694 (1.7)	31750 (1.6)		0.51	0.51
NLR (pre)[b]	2.45	1.033	*V*1		0.309	0.196		1.342	1.229		32087 (0.5)	32282 (0.1)		0.69	0.67
			*V*2b		0.269	0.183		1.302	1.216		32237 (0.1)	32282 (0.1)		0.66	0.67
			*V*3		0.304	0.224		1.337	1.257		32244 (0.0)	32282 (0.1)		0.66	0.67

The following cutoffs were used: BF, $\varepsilon_S = 0.10$ Å^2; DF, $\varepsilon_V = 0.01$ Å^3 (NL); $\varepsilon_V = 0.03$ Å^3 (NLR).
[a] See eq. (A.5).
[b] Including CPU time for NL.

TABLE V.

SASA Calculations of Penicillopepsin (3app, United Atom, 2366 Atoms, Numerical SASA = 12723 Å^2).

Pre- or Postprocessing Step	p[a]	CPU Pre- or Postprocessing Step(s) (s)	CPU Gaussian Calculations (s)				Total CPU (s)			SASA (Å^2) (% Diff. from Numer. Result)			Ave. Abs. Atomic Error (Å^2)		
			Variation	GS	BF	DF	GS	BF	DF	GS	BF	DF	GS	BF	DF
GS (pre)	1.35	7318.07		0.27			7318.34			25855 (103.2)			7.32		
NL (pre)	3.70	0.28	*V*1		7774.37	205.46		7774.65	205.74		13865 (9.0)	16560 (30.2)		1.63	3.43
			*V*2a		941.17	42.03		941.45	42.31		13850 (8.9)	16693 (31.2)		1.63	3.48
NLR (pre)[b]	3.90	0.71	*V*1		62.11	14.68		62.82	15.39		15431 (21.3)	15389 (21.0)		2.55	2.61
			*V*2b		540.68	32.25		541.39	32.96		15430 (21.3)	15389 (21.0)		2.55	2.61
			*V*3		73.54	17.26		74.25	17.97		15430 (21.3)	15392 (21.0)		2.55	2.61
BAE (pre)[c]	3.90	0.81	*V*1		23.58	5.96		24.39	6.77		13800 (8.5)	13820 (8.6)		1.90	1.98
			*V*2b		424.51	30.03		425.32	30.84		13768 (8.2)	13814 (8.6)		1.90	1.98
			*V*3		53.00	15.59		53.81	16.40		13782 (8.3)	13820 (8.6)		1.90	1.98
Fitting (post)[d]	3.05	0.81	*V*1		24.10	6.66		24.91	7.47		13665 (7.4)	13521 (6.3)		1.14	1.16

Preprocessing steps: NL, NLR, and BAE. The following cutoffs were used: BF, $\varepsilon_S = 0.5$ Å^2; DF, $\varepsilon_V = 0.15$ Å^3.

[a] See eq. (A.5).

[b] Including CPU time for NL.

[c] Including CPU time for NL and NLR.

[d] Using atom type dependent parameters.

TABLE VI.
Fitting Constants.

Atom Type, No. Bonded Neighbors	No. in All Test Compounds	vdWSA Fitting Constant[a]	vdWSA RMS Error	SASA Fitting Constant[b]	SASA RMS error
C *sp*3, 1	858	1.031	0.27	1.047	2.37
C *sp*3, 2	2069	1.040	0.37	0.860	3.16
C *sp*3, 3	2637	0.980	0.43	0.389	1.71
C *sp*3, 4	11	0.656	0.11	1.000[c]	—[c]
C *sp*2, 2	1009	1.042	0.38	0.777	2.62
C *sp*2, 3	2750	0.911	0.34	0.169	0.82
O *sp*3, 1	523	1.030	0.46	1.090	2.45
O *sp*3, 2	732	1.082	0.38	0.659	2.28
O *sp*2, 1	2187	1.055	0.46	1.073	2.46
O^- carboxylate, 1	347	1.054	0.39	1.124	2.30
N *sp*2, 1	459	1.035	0.46	1.058	3.08
N *sp*2, 2	2017	1.033	0.42	0.610	2.43
N *sp*2, 3	289	0.843	0.26	0.113	0.47
N *sp*3, 1	77	1.022	0.42	0.986	3.43
N *sp*3, 2	20	1.033	0.28	0.656	4.73
N *sp*3, 3	6	0.862	0.39	0.059	0.54
S, 1	5	1.004	0.19	0.948	2.57
S, 2	59	1.003	0.53	0.923	2.36
P, 3	7	0.902	0.57	0.449	5.23
P, 4	237	0.771	0.30	0.069	0.62
Cl, 1	5	0.998	0.35	1.078	1.36

[a] DF, V2a, NL, $p = 2.60$, $\varepsilon_V = 0.04$ Å^3.
[b] DF, V1, NL, NLR, BAE, $p = 3.05$, $\varepsilon_V = 0.15$ Å^3.
[c] All atoms are buried.

TABLE VII.
Gaussian vdWSA.

Compound	No. Atoms	Numer. vdWSA (Å^2)	Gaussian vdWSA (Å^2)	Aver. Abs. Atomic vdWSA Error (Å^2)	Max. Abs. Atomic Error (Å^2)	CPU (s)[a]
chb	11	159.3	159.1 (0.2%)	0.16	0.4	0.000
ctc	33	394.7	387.9 (1.7%)	0.52	1.7	0.002
sip	37	488.7	488.7 (0.0%)	0.30	1.1	0.002
nmx	51	672.6	668.1 (0.7%)	0.30	0.9	0.002
1van	121	1505.2	1506.2 (0.1%)	0.32	1.3	0.004
1crn	327	4252.1	4250.8 (0.0%)	0.27	1.4	0.011
103d	500	5762.0	5669.3 (1.6%)	0.31	1.8	0.020
2ins	770	9840.8	9847.8 (0.1%)	0.34	2.0	0.026
163d	813	9519.1	9454.6 (0.7%)	0.32	2.1	0.032
1lz1	1029	13270.1	13295.0 (0.2%)	0.28	1.4	0.033
2stw	1488	18121.9	18180.4 (0.3%)	0.30	2.1	0.052
2tra	1544	17545.4	17591.2 (0.3%)	0.31	2.0	0.065
1sbg	1559	20483.7	20493.0 (0.1%)	0.28	1.9	0.047
5tra	1821	20075.0	20105.9 (0.2%)	0.32	1.8	0.079
1inc	1846	23909.9	23929.3 (0.1%)	0.29	2.1	0.058
1kvd	1988	25843.6	25853.9 (0.0%)	0.29	1.7	0.061
3app	2366	30404.2	30429.7 (0.1%)	0.30	2.1	0.075

[a] Gaussian computation only.

TABLE VIII.
Gaussian SASA.

Compound	No. Atoms	Numer. SASA (Å²)	Gaussian SASA (Å²)	Aver. Abs. Atomic SASA Error (Å²)	Max. Abs. Atomic Error (Å²)	CPU (s)[a]
chb	11	315.9	311.2 (1.5%)	2.35	5.7	0.00
ctc	33	608.4	615.9 (1.2%)	2.09	9.8	0.15
sip	37	730.3	723.6 (0.9%)	2.03	9.1	0.12
nmx	51	906.9	906.2 (0.1%)	1.86	5.4	0.11
1van	121	1495.5	1557.8 (4.2%)	1.91	12.4	0.68
1crn	327	2976.3	3051.0 (2.5%)	1.60	9.7	1.11
103d	500	4426.8	4439.8 (0.3%)	1.25	8.6	2.41
2ins	770	5740.4	6006.2 (4.6%)	1.64	15.0	2.24
163d	813	5636.6	5890.2 (4.5%)	1.38	13.9	3.21
1lz1	1029	6739.9	7088.3 (5.2%)	1.42	11.1	3.01
2stw	1488	12266.9	13298.9 (8.4%)	2.01	14.0	5.11
2tra	1544	12507.3	12900.0 (3.1%)	1.45	12.1	6.14
1sbg	1559	9540.2	10063.7 (5.5%)	1.35	11.3	4.03
5tra	1821	14830.5	15070.9 (1.6%)	1.44	16.3	8.00
1inc	1846	10666.1	11284.5 (5.8%)	1.19	11.8	5.58
1kvd	1988	10934.0	11845.9 (8.3%)	1.26	16.3	5.40
3app	2366	12722.8	13521.2 (6.3%)	1.16	13.0	6.66

[a] Gaussian computation only.

putation using the GS preprocessing step is the fastest of all, but the preprocessing step itself is much slower than all others used. Overall, when GS is performed, the computation is 5 times slower than is DF-2a, even though the former spends only half the time computing Gaussians as does the latter. The accuracy obtained from GS was slightly lower than that obtained with the other methods.

The all-atom vdWSA results show the same trends (Table IV). DF-2a is still fastest and is now 10 times faster than GS overall, because the ratio of preprocessing CPU time to subsequent surface computation for GS is higher than in the united-atom case. The time spent computing Gaussians, however, is even further improved by GS when compared to the united-atom situation: we now spend only one-third the time doing this than we do in the best other method. GS accuracy continues to be mildly degraded. Introduction of NLR speeds up variations 1 and 2 but (as for united atoms) uses more CPU time than it saves subsequently. Also, accuracy is further degraded with NLR. Therefore, summarizing the vdWSA results, for either the united-atom or the all-atom representation, DF-2a, foregoing NLR and (as stated earlier) BAE, gives both optimal speed and optimal accuracy.

Optimal parameters are shown in Table VI. Table VII shows united-atom vdWSA results for all test compounds using these parameters. The fitting procedure improves the accuracy by a factor of 2 in for united-atom 3app without any additional CPU cost. The molecular vdWSA for compounds in the test set compounds differs from the numerical results by 0.0–1.7%. Grant and Pickup published relative errors[22] about 10 times larger than ours (1.5 vs. 0.0% for 1crn; 0.9 vs. 0.1% for 2ins; 1.1 vs. 0.1% for 3app).

GAUSSIAN SASA

The calculation of SASA with the Gaussian method is known to be difficult because of the large number of heavily overlapping atomic areas involved. Grant and Pickup pointed out that the single Gaussian representation leads to accurate measures only in the limit of weakly overlapping spheres.[22] Their results for strongly overlapping spheres, as in SASA, were not as good.

We found that the higher hard-sphere overlaps characteristic of SASA can be handled with a single Gaussian representation if the Gaussians are made narrower with respect to the hard spheres than is optimal for vdWSA. In other words, we found it beneficial to use a higher Gaussian p parameter [eq. (A.5), Appendix A] than that used for vdWSA. The optimal value of p was found to be 3.70 when using a full neighbor list (NL)

and 3.90 in a reduced neighbor list (NLR). For vdWSA/NL, the optimal value was 2.60.

For SASA, in contrast to vdWSA, we observe significant differences in speed and accuracy among the various methods. As for vdWSA, the GS preprocessing step affords the fastest Gaussian calculations, here by a factor of 22 over the best other method (Table V). As for vdWSA, however, the cost of GS makes this method the slowest overall. The GS preprocessing step requires more than 2 CPU h for 3app. Atomic and overall error are significantly worse when GS is used than with the other methods. For example, the computed molecular Gaussian surface area, when GS is applied as a preprocessing step, is more than twice the true numerical hard-sphere value obtained from MacroModel/BatchMin. Because hard-sphere areas computed using the GS preprocessing procedure are highly accurate, the reason for the loss of accuracy when using it to compute Gaussian surfaces cannot lie in the neglect of intersections of order exceeding eight. Rather, it must be that the Gaussian method, when it is carried out to include the higher order intersections as it is with the other algorithms, compensates for errors it makes in the computation of the intersections of lower order.

As with vdWSA, the DF method is always faster than the BF method for each variation. In contrast with vdWSA, however, introduction of NLR not only pays for itself but also has a large effect on speed: BF-1 becomes 125 times faster and DF-1 14 times faster when NLR is introduced. For SASA, unlike vdWSA, variation 1 is faster than variation 2 when using NLR, and the NLR overhead is negligible in comparison to the subsequent savings. Variation 3 is comparable in speed to variation 1. Introduction of NLR decreases the accuracy when using BF ($\varepsilon_S = 0.5$ Å^2) but increases it when using DF ($\varepsilon_V = 0.15$ Å^3).

Introduction of BAE as an additional preprocessing step speeds up all the algorithms, but especially variation 1. BAE defines as buried 932 (39%) of the 2366 atoms in 3app, but the DF-1 computation is then faster by a factor of 2.5 rather than the expected linear factor of 1.6. This superlinear speed-up presumably comes from the fact that buried atoms have longer neighbor lists than exposed atoms. Thus, BAE eliminates the computation for just those atoms whose surfaces would otherwise be the most expensive to compute.

Variation 2 and variation 3 show much smaller decreases of CPU time than variation 1 when BAE is used. This is because in these variations, BAE does not avoid the computation of some Gaussian volume intersections; only the computation of buried-atom surfaces are avoided. In variation 1, on the other hand, intersection volumes for buried atoms are skipped completely.

Introducing NLR and BAE into DF-1 makes the method faster by a factor of 30 over DF-1/NL and also improves the accuracy from 3.43 to 1.98 Å^2. DF-1/NLR/BAE is 4 times faster than the BF-1 with the same preoptimizations. Previously reported Gaussian surface computations[22–24] used a BF algorithm without NLR. DF-1/NLR/BAE is faster (by a factor of about 140) than BF-2/NL, the fastest such algorithm we found. For these reasons, optimal parameters for DF-1/NLR/BAE were used in computing SASA for all test compounds.

A p parameter of 3.05 gave the best accuracy after fitting. The DF volume cutoff, ε_V, was chosen to be 0.15 Å^3 because lower values gave only marginally better accuracies but significantly higher CPU times. For example, ε_V values of 0.20, 0.15, 0.10, 0.05, and 0.01 Å^3 gave accuracies of 1.23, 1.16, 1.15, 1.17, and 1.14 Å^2 but CPU times of 6.0, 6.7, 7.2, 8.4, and 10.7 s for Gaussian computations, respectively.

The effect of simultaneously optimizing p and a set of atom type dependent multipliers improves the accuracy to 1.16 Å^2. This comes at a cost of about 0.7 s of CPU time; the additional CPU cost arises from the fact that when fitting is introduced, the optimal value of p goes from 3.90 to 3.05. That is, the Gaussians become wider, which gives rise to more terms in eq. (1) that are above the ε_V cutoff.

Table VIII shows the final SASA values for all test compounds. The molecular SASA values differ from the numerical results by 0.1–8.4%; the average absolute atomic errors range from 1.16 to 2.35 Å^2, and the maximum atomic errors range from 5.4 to 16.3 Å^2.

A recently published exact analytical SASA method (GETAREA)[18] computes atomic surfaces and first derivatives with respect to Cartesian coordinates of atomic centers for a 2325-atom protein in 2.10 s; the same article presents results for the approximate methods MSEED[8] (1.30 s) and SASAD[14] [1.13 s as SASAD(4, 12) and 1.47 s as SASAD(4, 24), a higher level of accuracy], all on a platform very similar to ours. MSEED and SASAD are accurate to about 0.4 Å^2. Our recently published LCPO method[21] computes SASA values for 3app in 0.92 s (about 7 times faster than our best Gaussian results), including overhead for the same optimizations (NL, NLR, and BAE) and using the

same atomic radii. LCPO is less accurate than MSEED and SASAD, and this can be seen as a classic trade-off between accuracy and speed. The Gaussian method reported here, on the other hand, is more accurate than LCPO (for 3app, 1.2 vs. 1.5 Å^2 average and 13.0 vs. 14.3 Å^2 maximum absolute atomic error) but less accurate than the others and not as fast as any of them. Although we did not code derivatives, the Gaussian method readily affords analytical first and second derivatives.[22] Our extension of this method to SASA would thus appear to be the most accurate SASA method for which second derivatives have been published.

Discussion

In every algorithmic variation we have tried, DF search of the intersection volumes is faster than the BF search previously reported.[22] Although the difference is minimal when vdWSA is being computed, it is dramatic when SASA is computed.

Gibson and Scheraga's preprocessing method (GS) for the elimination of intersection volumes[27] provides the fastest subsequent Gaussian calculations for SASA by a factor exceeding 20 compared to the method that is fastest overall; however, the preprocessing step itself is very slow. Application of GS to the Gaussian method also leads to poor accuracy. Cancellation of error between high- and low-order Gaussian terms in the other methods probably contributes to their greater accuracy. However, if ways could be found to make the GS step faster, it could lead to a considerably faster overall method than any discussed here. Such an improvement would similarly speed up the subsequent computation of hard-sphere surfaces, the application originally published by the authors.[27]

Gaussian SASA can be computed quickly and accurately by introducing the two preprocessing steps of NLR[25] and BAE[26] in combination with the DF-1 method. Gaussian SASA accuracies are intermediate in the range of recently published methods, but the Gaussian method is somewhat slower; on the other hand, it does provide second derivatives as shown previously.[22]

The introduction of DF-1, NLR, and BAE speeds up Gaussian SASA computation by an overall factor of 1150 over the BF-1/NL algorithm. That such a great speedup is possible for SASA is due in part to the fact that the simple SASA Gaussian computation requires consideration of intersections of order up to 12. There are so many such intersections that great scope exists for savings if they can be explored in a clever, rather than a naive, manner. This is not the situation for vdWSA, which requires intersections of maximum order of only 6.

When SASA (rather than vdWSA) is computed, NLR reduces many more neighbors, because the pairwise overlaps are greater and the neighbor lists longer. Furthermore, BAE eliminates the surface computation for about half the atoms when SASA is computed but for none when vdWSA is computed. Finally, the use of a DF search (rather than BF search) improves efficiency by a factor of about four when NLR and BAE are used with SASA; for vdWSA, DF affords an improvement of only a few percent.

A programmer is often advised to ''Make it right before you make it faster.''[33] Although we concur with this advice as a matter of methodology, the work presented here provides at least a counterexample to the optimistic conclusion that because ''most programs spend 50 percent or more of their time in a very small portion (5 percent or so) of their code,''[34] the programmer intent on optimization need only identify and be concerned with this small portion. For SASA, DF-2 is the fastest method at the lowest (NL) optimization level explored. If a programmer had naively introduced NLR preoptimization and then made the minimal changes necessary to DF-2, the resulting algorithm would have been slower by a factor of 2.7 than the fastest (DF-1) NLR algorithm. If, still staying with DF-2, BAE was applied, a final 27% improvement over the original DF-2/NL result would have been achieved, considerably short of the sixfold improvement obtained by the use of DF-1/NLR/BAE. To achieve significant performance improvement, it was necessary to experiment with several algorithmic variants, each of which required major rethinking of data structures and code organization.

It is interesting to note that the algorithmic variant 1, which appears the most wasteful at first glance (because it computes identical Gaussian terms multiple times), is, in fact, the most efficient variant when the NLR and/or BAE preprocessing optimization is carried out.

The DF-1/NLR/BAE prescription is the best we have found for the computation of SASA using radial derivatives of the sum of signed intersection volumes [eq. (A.11), Appendix A]. Although we demonstrated this only for the Gaussian method, this conclusion should extend to other methods of computation based on this formula, such as hard-sphere surfaces. For Gaussian SASAs, this pre-

scription has the unusual property of optimizing both speed and accuracy.

Conclusion

The method reported here exhibits relative molecular vdWSA accuracies (relative to hard-sphere vdWSAs) about 10 times better than those reported earlier for the original Gaussian method.[22] Atomic vdWSAs are also computed extremely accurately; these quantities were not reported previously.

The successful accurate computation of Gaussian SASA was reported for the first time in this article. Although the Gaussian method has inherent deficiencies when highly overlapping spheres are encountered (as in SASA calculations), the results shown here demonstrate that these problems can be overcome and that the method can be used for reasonably accurate and fast computation of atomic and molecular SASA, even for large systems.

Acknowledgments

We wish to thank Dr. Ken D. Gibson and Prof. Harold A. Scheraga for making their Fortran code available to us. We would also like to thank Dr. Gibson, as well as Dr. Anthony Nicholls and Dr. Michael Beachy, for helpful discussions. We would like to thank the readers of alt.folklore.computers, particularly Tony Wingo, Michael F. Coyle, and Charles and Francis Richmond, for supplying refs. 33 and 34.

Appendix A

The spherical Gaussian of weight p_i is defined as

$$p_i^g(r_i) = p_i \cdot \exp(-\alpha r_i^2), \tag{A.1}$$

where the local radial variable

$$r_i = |r - R_i| \tag{A.2}$$

is defined with respect to the atomic center R_i as origin and the exponent

$$\alpha_i = \frac{\kappa_i}{\sigma_i^2}. \tag{A.3}$$

σ_i represents the vdW radius of atom i or the sum of the vdW radius and the radius of the solvent probe when SASA rather than vdWSA are computed. The dimensionless parameter κ_i is chosen such that

$$\kappa_i = \frac{\pi}{\lambda_i^{2/3}}. \tag{A.4}$$

The Gaussian weight is chosen such that

$$p_i \lambda_i = \frac{4\pi}{3}. \tag{A.5}$$

The Gaussian volume of a molecule can be written as

$$V^g = \sum_i V_i^g - \sum_{i<j} V_{ij}^g + \sum_{i<j<k} V_{ijk}^g - \sum_{i<j<k<l} V_{ijkl}^g + \cdots, \tag{A.6}$$

where the multiple summation terms represent the intersection volumes. According to the Gaussian product theorem,[22] the intersection volumes can be calculated as

$$V_{12\cdots n}^g = p^{12\cdots n} \cdot K^{12\cdots n} \cdot \left(\frac{\pi}{\Delta^{12\cdots n}}\right)^{3/2} \tag{A.7}$$

with the following definition of the above terms:

$$p^{12\cdots n} = \prod_{i=1}^{n} p_i, \tag{A.8}$$

$$K^{12\cdots n} = \exp\left[-\left(\frac{\sum_{i=1,\, j>i}^{n} \alpha_i \alpha_j \mathrm{dist}_{ij}^2}{\Delta^{12\cdots n}}\right)\right], \tag{A.9}$$

$$\Delta^{12\cdots n} = \sum_{i=1}^{N} \alpha_i. \tag{A.10}$$

The Gaussian atomic surface can be expressed as radial derivatives of the Gaussian volume intersections

$$A_m = \sum_i \frac{\partial V_i}{\partial \sigma_m} - \sum_{i<j} \frac{\partial V_{ij}}{\partial \sigma_m} + \sum_{i<j<k} \frac{\partial V_{ijk}}{\partial \sigma_m} - \sum_{i<j<k<l} \frac{\partial V_{ijkl}}{\partial \sigma_m} + \cdots \tag{A.11}$$

and the total surface area S of the molecule is therefore

$$S = \sum_{m=1}^{n} A_m. \tag{A.12}$$

The radial derivatives of the Gaussian intersection volumes are derived as

$$\frac{\partial V_{12\cdots n}^{g}}{\partial \sigma_i} = 2\frac{\kappa_i}{\sigma_i^3} V_{12\cdots n}^{g}\left(\frac{3}{2\Delta^{12\cdots n}} + R_i^{12\cdots n}\cdot R_i^{12\cdots n}\right), \quad \text{(A.13)}$$

where the vector

$$R_i^{12\cdots n} = R_i - \frac{\sum_{j=1}^{n}\alpha_j R_j}{\Delta^{12\cdots n}} \quad \text{(A.14)}$$

defines atom i with respect to the coalescence center of the intersecting spheres as the origin.

Appendix B

Given a candidate atom for BAE, four tetrahedral rays are defined as follows.[26] The rays start at the central atom and point in the directions of the vertices of a regular tetrahedron centered on the atom. The first ray points toward the atom's nearest neighbor; the second is as close as possible to the second nearest neighbor and the two others are then uniquely determined. In computing the four neighbor densities (NDs), each neighboring atom is assigned to the ray to which it lies closest. Conceptually, each ray forms the axis of a tetrahedral "cone" containing the atoms contributing to its ND.

ND $\rho_{i,k}$ for tetrahedral direction k of atom i is calculated as

$$\rho_{i,k} = \sum_{j}^{m_{i,k}} \exp\left(-\alpha_i \cdot \frac{d_{ij}^2}{(r_{H_2O} + r_j)^2}\right), \quad \text{(B.15)}$$

where $m_{i,k}$ is the number of neighbors of i in direction k; α_i is a constant that depends on the atom type of i; d_{ij} is the distance between atoms i and j; r_j is the vdW radius of neighboring atom j, and r_{H_2O} is the probe size of water (1.4 Å). An

TABLE B.I.
BAE Parameter.

Atom Type, No. Bonded Neighbors	α_i [a]	ρ_i^* [a]	Reorientation of Tetrahedrals?	Max. Atomic Error (Å²)	Ave. Abs. Atomic Error[b] (Å²)
C *sp*3, 1	0.57	1.93	Yes	3.7	0.20
C *sp*3, 2	0.59	2.25	No	3.3	0.15
C *sp*3, 3	0.56	2.23	No	4.0	0.04
C *sp*3, 4	[c]	[c]	[c]	[c]	[c]
C *sp*2, 2	0.50	2.38	Yes	3.2	0.25
C *sp*2, 3	0.63	1.46	Yes	3.8	0.10
O *sp*3, 1	0.78	1.42	Yes	1.1	0.05
O *sp*3, 2	0.68	1.43	No	1.5	0.19
O *sp*2, 1	0.68	1.96	Yes	2.1	0.05
O^- carboxylate, 1	[d]	[d]	[d]	[d]	[d]
N *sp*2, 1	[d]	[d]	[d]	[d]	[d]
N *sp*2, 2	0.70	1.20	Yes	4.1	0.10
N *sp*2, 3	0.64	0.99	Yes	1.7	0.07
N *sp*3, 1	[d]	[d]	[d]	[d]	[d]
N *sp*3, 2	[e]	[e]	[e]	[e]	[e]
N *sp*3, 3	[e]	[e]	[e]	[e]	[e]
S, 1	[e]	[e]	[e]	[e]	[e]
S, 2	0.77	1.74	No	1.1	0.09
P, 3	[d]	[d]	[d]	[d]	[d]
P, 4	[d]	[d]	[d]	[d]	[d]
Cl, 1	[e]	[e]	[e]	[e]	[e]

[a] See eq. (A.15).
[b] Calculated as follows: SASA of all BAE atoms / number of BAE atoms.
[c] Always buried, no special parameters needed.
[d] Less than 10% of atoms are buried; BAE is not performed.
[e] Not enough data for BAE parameter development.

atom is defined as buried if all four tetrahedral NDs $\rho_{i,k}$ are above a certain limit ρ_i^*, which is characteristic of atom type. The parameterization of α_i and ρ_i^* (Table B.I) for the given vdW radii (Table II) and solvent probe size is based on the same compounds and principles as described in ref. 26.

References

1. Lee, B.; Richards, F. M. J Mol Biol 1971, 55, 379–400.
2. Hermann, R. B. J Phys Chem 1972, 76, 2754–2759.
3. Wodak, S. J.; Janin, J. Proc Natl Acad Sci USA 1980, 77, 1736–1740.
4. Richmond, T. J. J Mol Biol 1984, 178, 63–89.
5. Hasel, W.; Hendrickson, T. F.; Still, W. C. Tetrahedron Comput Methodol 1988, 1, 103–116.
6. Dodd, L. R.; Theodorou, D. N. Mol Phys 1991, 72, 1313–1345.
7. Wesson, L.; Eisenberg, D. Protein Sci 1992, 1, 227–235.
8. Perrot, G.; Cheng, B.; Gibson, K. D.; Vila, J.; Palmer, K. A.; Nayeem, A.; Maigret, B.; Scheraga, H. A. J Comput Chem 1992, 13, 1–11.
9. von Freyberg, B.; Braun, W. J Comput Chem 1993, 14, 510–521.
10. Eisenhaber, F.; Argos, P. J Comput Chem 1993, 14, 1272–1280.
11. Mumenthaler, C.; Braun, W. J Mol Model 1995, 1, 1–10.
12. Kurochkina, N.; Lee, B. Protein Eng 1995, 8, 437–442.
13. Liotard, D. A.; Hawkins, G. D.; Lynch, G. C.; Cramer, C. J.; Truhlar, D. G. J Comput Chem 1995, 16, 422–440.
14. Sridharan, S.; Nicholls, A.; Sharp, K. A. J Comput Chem 1995, 16, 1038–1044.
15. Sanner, M. F.; Olson, A. J.; Spehner, J.-C. Biopolymers 1996, 38, 305–320.
16. Gabdoulline, R. R.; Wade, R. C. J Mol Graph 1996, 14, 341–353.
17. Cossi, M.; Mennucci, B.; Cammi, R. J Comput Chem 1996, 17, 57–73.
18. Fraczkiewicz, R.; Braun, W. J Comput Chem 1998, 19, 319–333.
19. Cui, Y.; Chen, R. S.; Wong, W. H. Proteins 1998, 31, 247–257.
20. Street, A. G.; Mayo, S. L. Folding & Design 1998, 3, 253–258.
21. Weiser, J.; Shenkin, P. S.; Still, W. C. J Comput Chem 1999, 20, 217–230.
22. Grant, J. A.; Pickup, B. T. J Phys Chem 1995, 99, 3503–3510.
23. Grant, J. A.; Gallardo, M. A.; Pickup, B. T. J Comput Chem 1996, 17, 1653–1666.
24. Dudek, M. J.; Ramnarayan, K.; Ponder, J. W. J Comput Chem 1998, 19, 548–573.
25. Weiser, J.; Weiser, A. A.; Shenkin, P. S.; Still, W. C. J Comput Chem 1998, 19, 797–808.
26. Weiser, J.; Shenkin, P. S.; Still, W. C. J Comput Chem 1999, 20, 586–596.
27. Gibson, K. D.; Scheraga, H. A. Mol Phys 1987, 62, 1247–1265.
28. Stouten, P. F. W.; Frömmel, C.; Nakamura, H.; Sander, C. Mol Simul 1993, 10, 97–120.
29. Augspurger, J. D.; Scheraga, H. A. J Comput Chem 1996, 17, 1549–1558.
30. Bernstein, F. C.; Koetzle, T. F.; Williams, G. J. B.; Meyer, E. F., Jr.; Brice, M. D.; Rodgers, J. R.; Kennard, O.; Shimanouchi, T.; Tasumi, M. J Mol Biol 1977, 112, 535–542. The Brookhaven Protein Data Bank is accessible via the internet at http://pdb.pdb.bnl.gov/.
31. Weiser, J.; Holthausen, M. C.; Fitjer, L. J Comput Chem 1997, 18, 1264–1281. Supplementary Material including data for sipholenol-A monoacetate is available via the internet at http://journals.wiley.com/0192-8651/wilma/wilma.cgi/v18.1264.html.
32. Mohamadi, F.; Richards, N. G. J.; Guida, W. C.; Liskamp, R.; Lipton, M.; Caufield, C.; Chang, G.; Hendrickson, T.; Still, W. C. J Comput Chem 1990, 11, 440–467. We used MacroModel Version 6.5.
33. Kernighan, B. W.; Plauger, P. J. The Elements of Programming Style; 2nd ed.; McGraw–Hill: New York, 1978; p 124.
34. Johnson, S. C.; Kernighan, B. W. Byte 1983, 8 (August), 48–60.